Package ‘genderizeR’

August 29, 2016

Type Package
Title Gender Prediction Based on First Names
Version 2.0.0
Date 2016-05-11
Description Utilizes the ‘genderize.io’ Application Programming Interface to predict gender from first names extracted from a text vector. The accuracy of prediction could be controlled by two parameters: counts of a first name in the database and probability of prediction.
License MIT + file LICENSE
URL https://github.com/kalimu/genderizeR,
http://www.wais.kamil.rzeszow.pl/genderizeR
BugReports https://github.com/kalimu/genderizeR
Imports stringr (>= 1.0.0), httr (>= 1.1.0), tm (>= 0.6-2), data.table (>= 1.9.6), magrittr, parallel (>= 3.3.0), utils
Depends R (>= 3.3.0)
Encoding UTF-8
LazyData true
Suggests testthat, knitr, rmarkdown
RoxygenNote 5.0.1
VignetteBuilder knitr
NeedsCompilation no
Author Kamil Wais [aut, cre],
Nathan VanHoudnos [ctb],
John Ramey [ctb]
Maintainer Kamil Wais <kamil.wais@gmail.com>
Repository CRAN
Date/Publication 2016-05-11 12:06:34
R topics documented:

authorships ................................................................. 2
classificationErrors .................................................... 3
classificationErrors .................................................... 4
findGivenNames .......................................................... 4
genderize ................................................................. 6
genderizeAPI ............................................................ 7
genderizeBootstrapError ................................................ 8
genderizePredict ....................................................... 10
genderizeR .............................................................. 10
genderizeTrain ......................................................... 11
givenNamesDB_authorships .......................................... 12
givenNamesDB_titles .................................................. 13
numberOfNames .......................................................... 14
textPrepare ............................................................. 14	
titles ..................................................................... 15

Index 16

authorships  Authorships sample

Description

A dataset containing a simple random sample of authorships (unique combination of authors and titles) from WebOfScience records of articles of "biographical-items" or "items-about-individual" types from all fields of study published from 1945 to 2014. The sample was drawn in December 2014.

Usage

authorships

Format

A data frame with 2641 rows and 5 variables:

title title of an article
authors all authors for this article
value a single author of the article - with the title forms an authorship
genderCoded manually coded gender of an author. There are four codes: "female", "male", "noname", "unknown". "Noname" is the code for a case where human coders were not be able to find a proper first name of an author. "Unknown" if the code for a case were the coders found a full name of an author but were not able to verify if she or he is a man or a female.

WOSaccessionNumber original ID of an article
classificationErrors

Source

http://webofknowledge.com/

classificationErrors  Calculating classification errors and other prediction indicators

Description

classificationErrors builds confusion matrix from manually coded and predicted gender vectors and returns different specific classification errors calculated on that matrix.

Usage

classificationErrors(labels, predictions)

Arguments

- **labels**: A vector of true labels. Should have following values: c("female", "male", "unknown", "noname"). *noname* stands also for initials only.

- **predictions**: A vector of predicted gender. Should have following values: c("female", "male", NA). *NA* when it was not possible to predict any gender.

Value

A list of gender prediction efficiency indicators:

- **confMatrix**: full confusion matrix

- **errorTotal**: total classification error

- **errorFullFirstNames**: classification error without "noname" category

- **errorCoded**: classification error without both "noname" and "unknown" category

- **errorCodedWithoutNA**: classification error only on "female" and "male" categories in both predictions and labels

- **naTotal**: total proportion of items with unpredicted gender

- **naFullFirstNames**: proportion of items with unpredicted gender without "noname" category

- **naCoded**: proportion of items with unpredicted gender without both "noname" and "unknown" category

- **errorGenderBias**: "male" classified as "female" minus "female" classified as "male" and divided by the sum of items in "female" and "male" categories in both predictions and labels
**Examples**

```r
## Not run:

set.seed(23)
labels = sample(c("female", "male", "unknown", "noname"), 100, replace = TRUE)
predictions = sample(c("female", "male", NA), 100, replace = TRUE)
classificationErrors(labels, predictions)

## End(Not run)
```

---

**classificationErrors**  
*Calculating classification errors and other prediction indicators*

**Description**

The `classificationErrors` function was misspelled (sorry for that!). Now the function has the proper name `classificationErrors` (with "i"). Old function name still works, but is deprecated now and will be removed in future version of the package.

**Usage**

`classificationErrors(labels, predictions)`

**Arguments**

- `labels`: A vector of true labels. Should have following values: c("female", "male", "unknown", "noname"). *noname* stands also for initials only.
- `predictions`: A vector of predicted gender. Should have following values: c("female", "male", NA). NA when it was not possible to predict any gender.

---

**findGivenNames**  
*Getting gender prediction data for a given text vector.*

**Description**

`findGivenNames` extracts from text unique terms and gets the gender prediction for all these terms.

**Usage**

```r
findGivenNames(x, textPrepare = TRUE, apikey = NULL, queryLength = 10, 
progress = TRUE, ssl.verifypeer = TRUE)
```
findGivenNames

Arguments

- **x**: A text vector or a character vector of unique terms prepared beforehand.
- **textPrepare**: If TRUE (default) the textPrepare function will be used on the x vector. Set it to FALSE if you already have prepared a character vector of cleaned up and deduplicated terms that you want to send to the API for first name gender checking.
- **apikey**: A character string with the API key obtained via https://store.genderize.io. A default is NULL, which uses the free API plan. If you reached the limit of the API you can start from the last checked term next time.
- **queryLength**: How much terms can be checked in a one single query
- **progress**: If TRUE (default) progress bar is displayed in the console
- **ssl.verifypeer**: Checks the SSL Certificate. Default is TRUE. You may set it to FALSE if you encounter some errors that break the connection with the API (though it is not recommended).

Value

A data table with given names found in database, gender predictions, probabilities of gender predictions, and counts how many people with a given name is recorded in the database.

Examples

```r
## Not run:

x = "Tom did play hookey, and he had a very good time. He got back home barely in season to help Jim, the small colored boy, saw next-day's wood and split the kindlings before supper—at least he was there in time to tell his adventures to Jim while Jim did three-fourths of the work. Tom's younger brother (or rather half-brother) Sid was already through with his part of the work (picking up chips), for he was a quiet boy, and had no adventurous, trouble-some ways. While Tom was eating his supper, and stealing sugar as opportunity offered, Aunt Polly asked him questions that were full of guile, and very deep—for she wanted to trap him into damaging revealments. Like many other simple-hearted souls, it was her pet vanity to believe she was endowed with a talent for dark and mysterious diplomacy, and she loved to contemplate her most transparent devices as marvels of low cunning. (from 'Tom Sawyer' by Mark Twain)"

xProcessed = textPrepare(x)

foundNames = findGivenNames(xProcessed, textPrepare = FALSE)
foundNames[count > 100]

# (the results can differ due to new, updated data pulled from the API)
# name gender probability count
# 1: jim male 1.00 2291
# 2: mark male 1.00 6178
# 3: polly female 0.99 191
```
genderize

Predicting gender for character strings.

Description

For each character string in x vector genderize use output of the findGivenNames function and returns a gender prediction for the whole character string based on possible first name terms located inside those strings.

Usage

genderize(x, genderDB, blacklist = NULL, progress = TRUE)

Arguments

x A vector of text strings.
genderDB A data.table output of findGivenNames function for the vector x.
blacklist Some terms could be excluded from gender checking
progress If TRUE (default) progress bar is displayed in the console

Value

A data table with text string, a term found in genderDB, that is finally used as a given name to predict gender, a predicted gender, number of potential gender indicators ("1" if only one term from the text string is found in genderDB).

Examples

## Not run:

```r
x = c("Winston J. Durant, ASHP past president, dies at 84", 
"Gold Badge of Honour of the DGAI Prof. Dr. med. Norbert R. Roewer Wuerzburg", 
"The contribution of professor Yu.S. Martynov (1921-2008) to Russian neurology", 
"JAN BASZKIEWICZ (3 JANUARY 1930 - 27 JANUARY 2011) IN MEMORIAM", 
"Maria Sklodowska-Curie")
givenNames = findGivenNames(x)
givenNames = givenNames[count>40]
genderize(x, genderDB=givenNames, blacklist=NULL)
```

# 4: text
# 1: text

### Not run:

```
# 4:   tom   male       1.00 3736

## End(Not run)
```
genderizeAPI

# 2: Gold Badge of Honour of the DGAI Prof. Dr. med. Norbert R. Roewer Wuerzburg
# 3: The contribution of professor Yu.S. Martynov (1921-2008) to Russian neurology
# 4: JAN BASZKIEWICZ (3 JANUARY 1930 - 27 JANUARY 2011) IN MEMORIAM
# 5: Maria Sklodowska-Curie

# givenName gender genderIndicators
# 1: winston male 1
# 2: med male 2
# 3: NA NA 0
# 4: jan male 1
# 5: maria female 1

## End(Not run)

genderizeAPI

**Getting data from genderize.io API**

Description

genderizeAPI connects with genderize.io API and checks if a term (one or more) is in the given names database and returns its gender probability and count of the cases recorded in the database.

Usage

genderizeAPI(x, apikey = NULL, ssl.verifypeer = TRUE)

Arguments

- **x**: A vector of terms to check in genderize.io database.
- **apikey**: A character string with the API key obtained via https://store.genderize.io. A default is `NULL`, which uses the free API plan.
- **ssl.verifypeer**: Checks the SSL Certificate. Default is `TRUE`.

Value

A list of four elements: `response` is a data frame with names, genders, probabilities and counts or `NULL` if none of the terms are not located in the genderize.io database; `limitLeft` is showing how many queries to the API are still possible within the current `limit` which will be renewed in `limitReset` seconds.

Examples

```r
# Not run:
terms = c("loremipsum")
genderizeAPI(terms)$response
# Null data.table (0 rows and 0 cols)
```
genderizeBootstrapError

Gender prediction errors on bootstrap samples

Description

genderizeBootstrapError calculates the Apparent Error Rate, the Leave-One-Out bootstrap error rate, and the .632+ error rate from Efron and Tibshirani (1997). The code is modified version of several functions from sortinghat package by John A. Ramey.

Usage

genderizeBootstrapError(x, y, givenNamesDB, probs, counts, num_bootstraps = 50, parallel = FALSE)

Arguments

x       A text vector that we want to genderize
y       A text vector of true gender labels for x vector
givenNamesDB A dataset with gender data (could be an output of findGivenNames function)
probs    A numeric vector of different probability values. Used to subsetting a given-NamesDB dataset

Example of the function output:

<table>
<thead>
<tr>
<th>name</th>
<th>gender</th>
<th>probability</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>jan</td>
<td>male</td>
<td>0.60</td>
<td>1692</td>
</tr>
<tr>
<td>maria</td>
<td>female</td>
<td>0.99</td>
<td>8467</td>
</tr>
<tr>
<td>norbert</td>
<td>male</td>
<td>1.00</td>
<td>77</td>
</tr>
<tr>
<td>winston</td>
<td>male</td>
<td>0.98</td>
<td>128</td>
</tr>
</tbody>
</table>

Gender prediction errors on bootstrap samples

## End(Not run)
genderizeBootstrapError

counts A numeric vector of different count values. Used to subsetting a givenNamesDB dataset
num_bootstraps Number of bootstrap samples. Default is 50.
parallel It is passed to genderizeTrain function. If TRUE it computes errors with the use of parallel package and available cores. Default is FALSE.

Value

A list of bootstrap errors:

apparent Apparent Error Rate
loo_boot LOO-Boot Error Rate
errorRate632plus .632+ Error Rate

See Also

In the sortinghat package: errorest_apparent errorest_loo_boot errorest_632plus.

Examples

## Not run:

x <- c('Alex', 'Darrell', 'Kale', 'Lee', 'Robin', 'Terry', rep('Robin', 20))
y <- c(rep('female', 6), rep('male', 20))
givenNamesDB = findGivenNames(x)
pred = genderize(x, givenNamesDB)
classificationErrors(labels = y, predictions = pred$gender)

classificationErrors(labels = y, predictions = pred$gender)

probs = seq(from = 0.5, to = 0.9, by = 0.05)
counts = c(1)

set.seed(23)
genderizeBootstrapError(x = x, y = y,
                        givenNamesDB = givenNamesDB,
                        probs = probs, counts = counts,
                        num_bootstraps = 20,
                        parallel = TRUE)

# $apparent
# [1] 0.9615385

# $loo_boot
# [1] 0.965812

# $errorRate632plus
# [1] 0.964225
genderizeR

---

**genderizePredict**  
*Gender predicting function*

**Description**

genderizePredict predicts gender with the best values of probability and count parameters.

**Usage**

genderizePredict(trainedParams, newdata, givenNamesDB)

**Arguments**

- `trainedParams`: An output of a `genderizetrain` function with prediction efficiency indicators for different combinations of probability and count values
- `newdata`: A character vector for gender prediction
- `givenNamesDB`: A dataset with gender data (could be an output of `findGivenNames` function)

**Value**

A character vector of values: male, female or unknown.

---

**genderizer**  
*Gender Prediction Based on First Names*

**Description**

The `genderizer` package uses genderize.io API to predict gender from first names extracted from text corpuses. The accuracy of prediction could be controlled by two parameters: counts of first names in database and probability of gender given the first name.

**Details**

If you need help with your research or commercial projects, feel free to contact me via my homepage contact form: [http://www.wais.kamil.rzeszow.pl/genderizer](http://www.wais.kamil.rzeszow.pl/genderizer)

**See Also**

- [http://www.wais.kamil.rzeszow.pl/genderizer](http://www.wais.kamil.rzeszow.pl/genderizer) [R package homepage]
- [https://github.com/kalimu/genderizeR](https://github.com/kalimu/genderizeR) [source code of the latest development version of the R package]
- [http://genderize.io/](http://genderize.io/) [Homepage of genderize.io API]
genderizeTrain

Training genderize function

description

genderizeTrain predicts gender and checks different combinations of probability and count parameters.

Usage

genderizeTrain(x, y, givenNamesDB, probs, counts, parallel = FALSE, cores = NULL)

Arguments

x A text vector that we want to genderize.
y A text vector of true gender labels for x vector.
givenNamesDB A dataset with gender data (could be an output of findGivenNames function).
probs A numeric vector of different probability values. Used to subsetting a givenNamesDB dataset.
counts A numeric vector of different count values. Used to subsetting a givenNamesDB dataset.
parallel If TRUE it computes errors with the use of parallel package and available cores. Default is FALSE.
cores A integer value for number of cores designated to parallel processing or NULL (default). If parallel argument is TRUE and cores is NULL, than the available number of cores will be detected automatically.

Value

A data frame with all combination of parameters and computed sets of prediction indicators for each combination:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorCoded</td>
<td>classification error for predicted &amp; unpredicted gender</td>
</tr>
<tr>
<td>errorCodedWithoutNA</td>
<td>classification error for predicted gender only</td>
</tr>
<tr>
<td>naCoded</td>
<td>proportion of items with manually codded gender and with unpredicted gender</td>
</tr>
<tr>
<td>errorGenderBias</td>
<td>net gender bias error</td>
</tr>
</tbody>
</table>

See Also

Examples

```r
## Not run:

x = c('Alex', 'Darrell', 'Kale', 'Lee', 'Robin', 'Terry', 'John', 'Tom')
y = c(rep('male', length(x)))

givenNamesDB = findGivenNames(x)
probs = seq(from = 0.5, to = 0.9, by = 0.1)
counts = c(1, 10)

genderizeTrain(x = x, y = y,
givenNamesDB = givenNamesDB,
probs = probs, counts = counts,
parallel = TRUE)

# prob count errorCoded errorCodedWithoutNA naCoded errorGenderBias
# 1: 0.5 1 0.125 0.125 0.000 0.125
# 2: 0.6 1 0.125 0.000 0.125 0.000
# 3: 0.7 1 0.125 0.000 0.125 0.000
# 4: 0.8 1 0.375 0.000 0.375 0.000
# 5: 0.9 1 0.500 0.000 0.500 0.000
# 6: 0.5 10 0.125 0.125 0.000 0.125
# 7: 0.6 10 0.125 0.000 0.125 0.000
# 8: 0.7 10 0.125 0.000 0.125 0.000
# 9: 0.8 10 0.375 0.000 0.375 0.000
# 10: 0.9 10 0.500 0.000 0.500 0.000

## End(Not run)
```

givenNamesDB_authorships

**Gender data for authorship sample**

Description

A dataset with first names and gender data from genderize.io for the sample of authorships in this package. This is the output of findGivenNames function that was performed on December 26, 2014.

Usage

givenNamesDB_authorships

Format

A data.table object with 872 rows and 4 variables:

- **name** first name
**givenNamesDB_titles**

- **gender** predicted gender
- **probability** how many persons in with this first name has the predicted gender
- **count** how many persons in the genderize.io database had that first name

**Source**

[http://genderize.io/](http://genderize.io/)

---

**givenNamesDB_titles  Gender data for titles sample**

**Description**

A dataset with a gender data from genderize.io for the sample of titles in this package. This is the output of `findGivenNames` function that was performed on December 26, 2014.

**Usage**

`givenNamesDB_titles`

**Format**

A data.table object with 872 rows and 4 variables:

- **name** first name
- **gender** predicted gender
- **probability** how many persons in with this first name has the predicted gender
- **count** how many persons in the genderize.io database had that first name

**Source**

[http://genderize.io/](http://genderize.io/)
**numberOfNames**

*Number of names in the database.*

**Description**

`numberOfNames` returns a number of distinct names in the genderize.io database scrapped from genderize.io page.

**Usage**

```r
numberOfNames()
```

**Value**

returns a numeric value

**Examples**

```r
## Not run:

numberOfNames()

## End(Not run)
```

---

**textPrepare**

*Preparing text vector for gender prediction*

**Description**

`textPrepare` takes a text vector and converts it into a vector of unique terms. This function is used by default by the `findGivenNames` function as text pre-processor before sending a query to the API.

**Usage**

```r
textPrepare(x, textPrepMessages = FALSE)
```

**Arguments**

- `x` A vector of character strings.
- `textPrepMessages`
  
  If TRUE verbose output of the preparing process is shown on the console.
Value

A vector of unique terms with at least two characters.

Examples

```r
# Not run:

x = c("Winston J. Durant, ASHP past president, dies at 84",
       "Gold Badge of Honour of the DGAi Prof. Dr. med. Norbert R. Roewer Wuerzburg",
       "The contribution of professor Yu.S. Martynov (1921-2008) to Russian neurology",
       "JAN BASZKIEWICZ (3 JANUARY 1930 - 27 JANUARY 2011) IN MEMORIAM",
       "Maria Sklodowska-Curie")

head(textPrepare(x))
# [1] "ashp"  "at"    "badge"   "baszkiewicz"
# [5] "contribution" "curie"

# End(Not run)
```

### Description

A dataset containing a simple random sample of article titles from WebOfScience records of articles of "biographical-items" or "items-about-individual" types from all fields of study published from 1945 to 2014. The sample was drawn in December 2014.

### Usage

```r
titles
```

### Format

A data frame with 2641 rows and 2 variables:

- **title**: title of an article
- **genderCoded**: manually coded gender of an author. There are four codes: "female", "male", "noname", "unknown". "Noname" is the code for a case were human coders were not be able to find a proper first name of an author. "Unknown" if the code for a case were the coders found a full name of an author but were not be able to verify if she or he is a man or a female.

### Source

http://webofknowledge.com/
Index

*Topic datasets
  authorships, 2
  givenNamesDB_authorships, 12
  givenNamesDB_titles, 13
  titles, 15

authorships, 2

classificationErrors, 3
classificationErrors, 4

errorest_632plus, 9
errorest_apparent, 9
errorest_loo_boot, 9

findGivenNames, 4

genderize, 6
genderizeAPI, 7
genderizeBootstrapError, 8
genderizePredict, 10
genderizeR, 10
genderizeR-package (genderizeR), 10
genderizeTrain, 11
givenNamesDB_authorships, 12
givenNamesDB_titles, 13

numberOfNames, 14

textPrepare, 14
titles, 15